H.E.B. Ecole Superieur d'Informatique

Laboratoire de C++ : projet 2

Starlight

Auteurs :
Paul Kriwin
Simon Placentino

Titulaire du cours :
Dr. Romain Absil

18 avril 2015

Table des matières

Chapitre 1

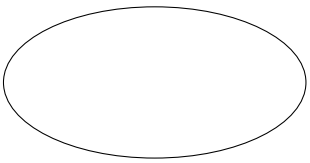
Introduction

Chapitre 2

Présentation succinte des classes

2.1 geometry

2.1.1 ellipse.hpp



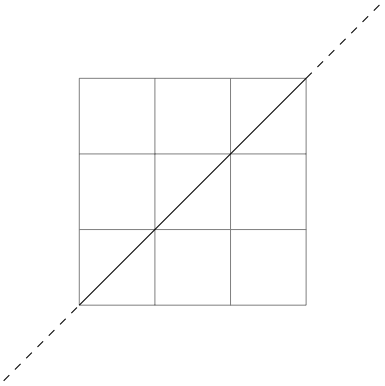
Une ellipse est un objet géométrique à deux dimensions représentée par une courbe plane fermée obtenu par découpe d'un cône sur un plan. Si ce dernier est perpendiculaire à l'axe du cône, l'ellipse sera alors un cercle. Une ellipse sera défini par trois éléments :

- une coordonnée cartésienne de son centre,
- une distance séparant le centre de l'intersection avec une paralelle à l'axe des ordonnées tangante à l'ellipse voulue,
- une distance sépérant le centre de l'intersection avec une paralelle à l'axe des abscisse tangante à l'ellipe voulue.

Ces éléments nous permettront de tracer une ellipse selon cette équation :

$$\frac{(x - c_x)^2}{{x_{radius}}^2} + \frac{(y - c_y)^2}{{y_{radius}}^2} = 1$$

2.1.2 line.hpp



Une droite est une ligne sans épaisseur, rectiligne et infinie dans le plan. Pour exister, une droite aura besoin :

- d'un coefficiant angulaire $m = \frac{\Delta y}{\Delta x}$ représentant la distance à parcourir sur l'axe des ordonnées pour une unité de distance sur l'axe des abcisses.
- d'un terme indépendant $p = \frac{y}{m \cdot x}$ représentant le décalage de chaque point sur l'axe des ordonnées,
- ou de deux points de coordonnées dans le plan,
- ou d'un point de coordonnées dans le plan et d'un coefficiant angulaire.

Ces éléments nous permettent de tracer une droite selon cette équaiton :

$$y = m \cdot x + p$$

2.1.3 rectangle.hpp

Un rectangle est une forme géométrique à 4 segments de droite [1] parallèle deux à deux.Ceux-ci vont donc former 4 angles droit ($\frac{\pi}{2} rad$) Cette forme peut être représentée par :

- la coordonnée du coin supérieur gauche $Sg = (x, y)$
- la grandeur des deux segments formant un angle de $\frac{\pi}{2} rad$ en ce point $hauteur$ et $largeur$.

Ainsi, il sera aisé de déterminer la position des autres coins

- $Sd = (Sg_x + largeur, Sg_y)$
- $Ig = (Sg_x, Sg_y + hauteur)$
- $Id = (Sg_x + largeur, Sg_y + hauteur)$

1. Un segment droite est une partie de droite délimitée par deux points non confondus

2.1.4 point.hpp

Un point est un objet mathématique permettant de situer un element dans un plan ou dans l'espace. Dans notre cas, plus spécifiquement dans un plan à deux dimensions. Celui-ci peut-être représenté de plusieurs manières dans le plan cartésien [2] :

- sous la forme d'une coordonnées cartésienne à l'aide de
  - une origine,
  - deux vecteurs partant de cette origine et perpendiculaires,
- et sous la forme d'une coordonnée polaire à l'aide de
  - une origine,
  - une coordonnée radiale r,
  - une coordonnée angulaire α.

2.1.5 utilities.hpp

La classe utilitaires mis en place ici est un ensemble de fonctions et valeurs constantes spécifiquement définies pour les calculs intervenant dans le projet.

constantes :

PI est un approximation de π sur 26 décimales,

PI_2 est une approximation de $\frac{\pi}{2}$ sur 26 décimales,

PI_4 est une approximation de $\frac{\pi}{4}$ sur 26 décimales,

EPSILON est une marge d'erreur de 10^{-7},

INF représente un nombre dit "infini" dans le milieu informatique.

fonctions :

fonction fonction

2. lien vers plan cartésien

2.2 elements

2.2.1 element.hpp

2.2.2 crystal.hpp

2.2.3 dest.hpp

2.2.4 lens.hpp

2.2.5 level.hpp

2.2.6 levelfactory.hpp

2.2.7 mirror.hpp

2.2.8 nuke.hpp

2.2.9 ray.hpp

2.2.10 source.hpp

2.2.11 wall.hpp

2.3 exception

2.3.1 starlightexception.hpp

Il est nécessaire, pour bon nombre des classes créées, de valider les arguments passés en paramètre dans le but de ne pas produire d'objets incohérents par rapport à l'analyse préalable du travail à fournir. Pour ce faire, des exceptions doivent être levées quand une instanciation créera un objet non désiré. Cette classe hérite de std : :exception appartenant à la librairie standard. Elle n'a aucune capacité supplémentaire mise à part être spécifique à ce projet.

Les différentes classes pouvant lever cette exception sont :

crystal si la taille de son rayon ne lui permet pas d'exister dans le plan,

lens si son interval de longueur d'onde n'est pas cohérent,

level si ses dimensions ne lui permettent pas d'exister dans le plan,

mirror si ses dimensions ne lui permettent pas d'exister dans le plan, si sa position ou son angle n'entre pas dans les limites imposées,

nuke si la taille de son rayon ne lui permet pas d'exister dans le plan,

ray si sa longueur d'onde n'entre pas dans l'interval cohérent imposé,

source si sa longueur d'onde n'entre pas dans l'interval cohérent imposé,

wall si ses points déterminants ne lui permettent pas d'exister dans le plan,

ellipse si ses dimensions ne lui permettent pas d'exister dans le plan,
rectangle si ses dimensions ne lui permettent pas d'exister dans le plan.

2.4 view

Chapitre 3

Structure générale du programme

Chapitre 4

Détail des algorithmes utilisés

4.1 Algorithme de réflection

4.2 Algorithme d'intersection

4.2.1 Intersection de deux droites

4.2.2 Intersection d'une droite et d'un rectangle

4.2.3 Intersection d'une droite et d'une ellipse

Chapitre 5

Test effectués

Chapitre 6

Conclusion

Annexe A

Réferences